Image Processing - CS 474
Programming Assignment 2
Anthony Silva
October 20, 2025

“I declare that all material in this assignment is my own work except where there is clear acknowledgment or reference to the work of others. I understand that both my report and code may be subjected to plagiarism detection software, and fully accept all consequences if found responsible for plagiarism, as explained in the syllabus, and described in UNR’s Academic Standards Policy: UAM 6,502.”

- Anthony Silva

Theory

Introduction

Spatial Filtering methods apply operations on a neighborhood of image pixels to produce a value for an output pixel to transform the original image in unique ways. Typically, this neighborhood is defined through a K x K window, where K is typically odd so it has a clear center. The operation itself can be linear or non linear, depending on the operator used in the operation on the input neighborhood. Common non-linear filters include max, min, median, and mode of the neighborhood. Linear operations typically involve weighted combinations of neighborhood pixels, commonly implemented through correlations or convolutions.

Correlations

A correlation operation on an image f(i, j) with a mask w(s, t) of size K x K is defined as with K being the size of the mask w centered on a pixel i + s and j + t on image f to produce the output g(i, j).

$$g(i,j) = w(s,t) \bullet f(i,j) = \sum_{s=-\lfloor K/2 \rfloor}^{\lfloor K/2 \rfloor} \sum_{t=-\lfloor K/2 \rfloor}^{\lfloor K/2 \rfloor} w(s,t) f(i+s, j+t)$$

Convolution Equation

A convolution is similar but involves rotation of the mask by 180 degrees. This is achieved by subtracting the mask indices s and t in the formula.

$$g(i,j) = w(s,t) * f(i,j) = \sum_{s=-\lfloor K/2 \rfloor}^{\lfloor K/2 \rfloor} \sum_{t=-\lfloor K/2 \rfloor}^{\lfloor K/2 \rfloor} w(s,t) f(i-s, j-t)$$

Correlation Equation

For symmetric masks, correlations and convolutions produce identical results. Correlations are particularly useful for pattern detection, as high output values indicate strong similarity between the mask and the corresponding image region.

Smoothing

Smoothing filters work as a low-pass filter, attenuating high-frequency components (edges, noise, fine details) while preserving low-frequency information (general shapes, gradual intensity variations).

Linear smoothing filters use masks where all elements are positive and normalized to sum to 1. The two most common types are the Averaging Filter, where all mask elements have an equal weight, and a Gaussian Filter, where elements are sampled from a 2D Gaussian distribution at

the mask center. The standard deviation of the gaussian distribution is typically tied to the size of the mask. Mask elements are normalized after sampling to ensure they sum to 1. Gaussian filters produce smoother results than averaging filters because they weigh center pixels more heavily, creating a gradual transition rather than an abrupt cutoff.

$$w(s,\ t)\ =\ \frac{1}{K^2}$$

Averaging Mask

$$G(x,\ y)\ =\ \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Gaussian Mask, where σ is the standard deviation controlling the spread.

Median Filtering is a non-linear operation that replaces each pixel with the median value of its neighborhood.

$$g(i,\ j)\ =\ median\{f(x\ +\ s,\ y\ +\ y):\ \forall s \forall t\ in\ the\ KxK\ window\}$$

Unlike averaging and gaussian methods, the median is robust to extreme outlier values like salt-and-pepper noise with either completely black or white values that damage the proper contrast of the image.

Sharpening Filters

Sharpening filters enhance edges and fine details by functioning as high-pass filters. They emphasize high-frequency components in the image.

Linear sharpening masks contain both positive and negative weights that sum to zero (after normalization). Unsharp masking is a linear sharpening mask that creates the mask by subtracting an original image with its blurred version (from a smoothing filter). Then the mask is added back to the original image to create a sharpened image. High boost filtering scales the mask with a constant k >= 1 to create the sharpened image. Higher k's produce sharper sharpening.

$$f_{sharp}(i,\ j)\ =\ f(i,\ j)\ +\ k\ \cdot\ [f(i,\ j)\ -\ f_{blur}(i,\ j)]$$

Derivatives

Derivatives of an image can be used to capture rate of change in pixel intensities, which makes them excellent for edge detection.

The first derivative, or the gradient, of an image is a vector containing the partial derivatives in x and y directions.

$$grad(f) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The gradient magnitude represents the strength of edges:

$$|grad(f)| = \sqrt{(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2}$$

The gradient direction shows the orientation of edges:

$$\theta = arctan(\frac{\partial f/\partial y}{\partial f/\partial x})$$

Certain masks can be used to approximate the partial derivatives, such as the prewitt and sobel masks.

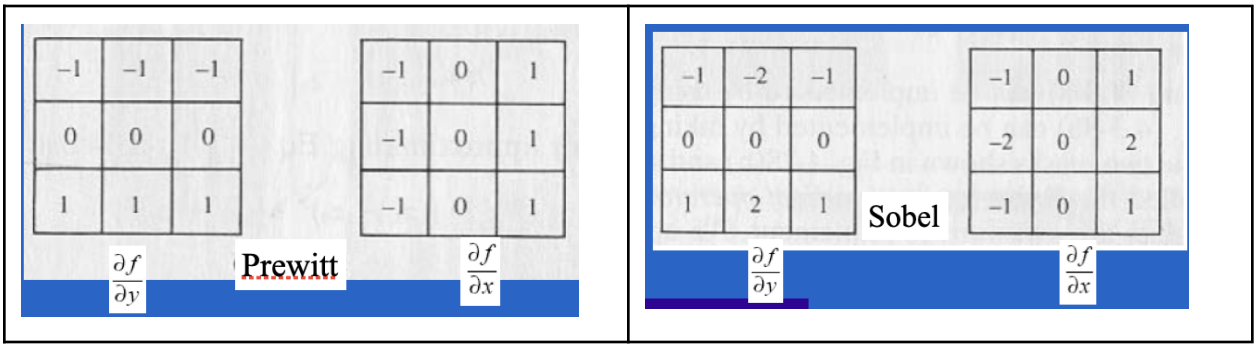


Table showing images of the common Prewitt and Sobel masks used to find gradients on an image in the x and y directions.

The Laplacian is the sum of second partial derivatives:

$$Laplacian(f) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

The Laplacian can also be estimated with masks:

0	1	0	1	1	1
1	−4	1	1	−8	1
0	1	0	1	1	1
0	−1	0	−1	−1	−1
−1	4	−1	−1	8	−1
0	−1	0	−1	−1	−1

Image showing four common masks to find the laplacian (second derivative) of the pixels in an image.

The Laplacian detects zero-crossings where the derivative changes sign, producing thinner edges than the gradient. However, it is more sensitive to noise due to its second-order nature. The Laplacian also does not provide directional information like the gradient does.

In this report, I implement some smoothing and sharpening operations in C++ to work on input images to see how they affect the input image and what their advantages and disadvantages are.

Results

Correlation

With my correlation implementation, I was able to create a mask (shown as ‘Correlation Mask’) that represented a graphic that appears in my source image multiple times (shown as ‘Image’). Running the mask over each pixel in the image, a resulting image was made, with brighter values correlating to sections in the image that ‘correlate’ better with the mask, i.e. that section of the image looks like the mask.

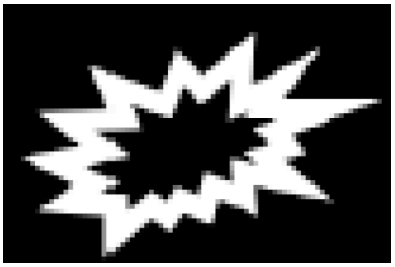
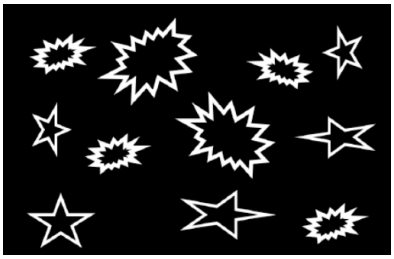
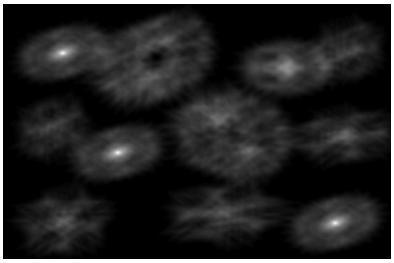
Correlation Mask	Image	Correlation Result
		

Table showing the correlation operation with the mask used, the image it was used on, and the result of the correlation operation on each pixel.

Smoothing

I implemented averaging and gaussian windows with sizes K = 7 and K = 15 to explore smoothing filters. In the table, the results of these various filters are shown.

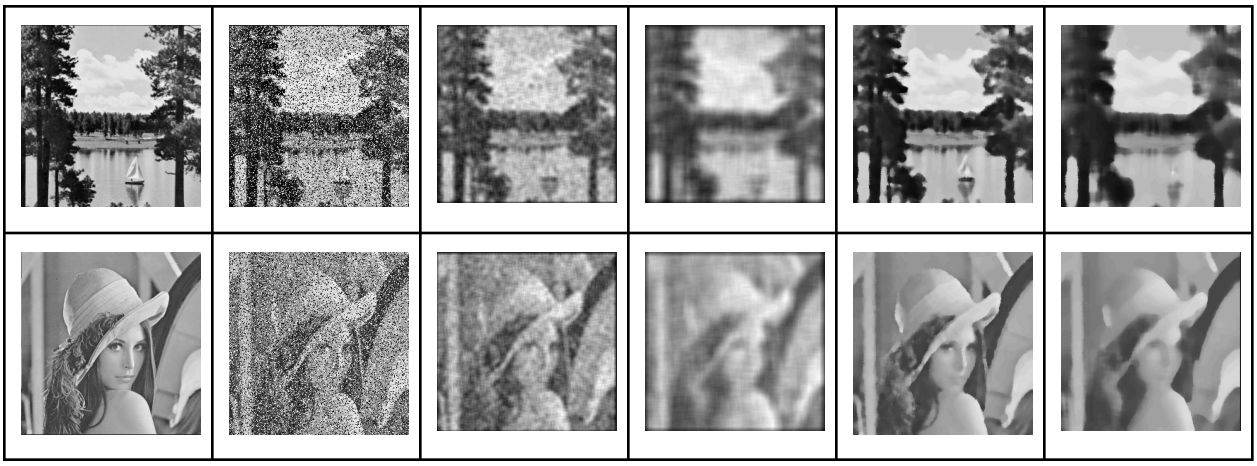
Source Image	Averaging K=7	Averaging K=15	Gaussian K=7	Gaussian K=15
				
				

Table showing averaging and gaussian operations on two source images with filter sizes 7 and 15.

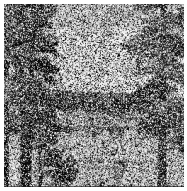
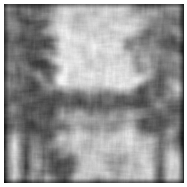
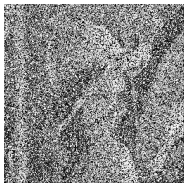
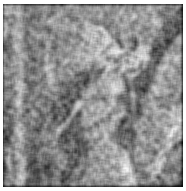
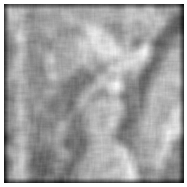
Median Filtering

Median Filtering is a non linear filter that works on a neighborhood by making the output pixel equal to the median of the input neighborhood. The use of the nonlinear median function results in slightly different behavior compared to something like an averaging filter. This difference in behavior is more evident when there are outliers involved (thinking of the fundamental differences in an average versus a median). To see these differences with outliers, the source images were randomly peppered with sharp noise of either pure black or white values and then the smoothing filters were applied to see how differently they remove noise. The experiment was performed in two rounds, one where images had 30 percent of their pixels replaced with noise and one with 50 percent. In each experiment, an averaging and median filtering filter was applied to each noised image with window sizes of 7 and 15. The results are seen below.

Original	With Noise (30 Percent)	Averaging Size 7	Averaging Size 15	Median Filtering Size 7	Median Filtering Size 15



Averaging and Median Filtering on Images with 30% Noise.

Original	With Noise (50 Percent)	Averaging Size 7	Averaging Size 15	Median Filtering Size 7	Median Filtering Size 15
					
					

Averaging and Median Filtering on Images on 50% Noise.

Derivatives

The gradient and laplacian masks were used to produce the resulting images from input images and the results of those masks are shown below. The magnitudes were also calculated for the prewitt and sobel masks to show the total change from the x and y directions in one output image.

Original	Laplacian	Prewitt df/dy	Prewitt df/dx	Sobel df/dy	Sobel df/dx

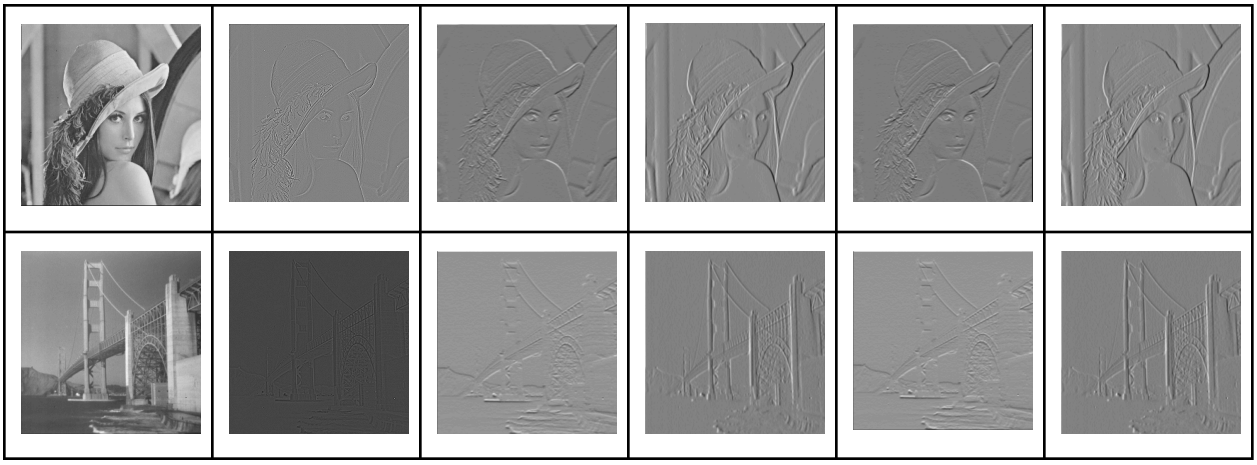


Table showing the output derivatives produced from the correlation of the derivative masks and the original image.

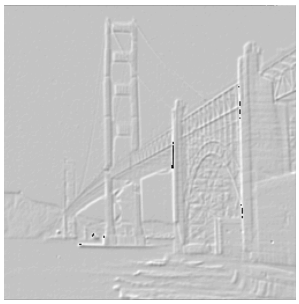
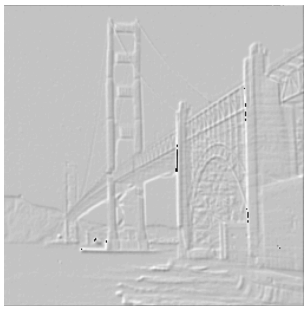
Sobel Magnitude	Prewitt Magnitude
	
	

Table showing the magnitudes of the gradient outputs for each mask on the original image.

Sharpening

Sharpening Filters were explored through the implementation of unsharp masking and high boost filtering. It is important to note that unsharp masking is a special case of high boost filtering where the scaling constant of the difference mask being added back is 1. The original image was smoothed with a 7x7 gaussian filter, and then the difference of the original and the smoothed image were used to create the “Difference Mask” in the table. This mask was added back at various constants and those results are shown below.

Original	Difference Mask	k=1 (Unsharp Masking)	k=1.5	k=2	k=3	k=5
						
						

Table showing the results of sharpening filters.

Discussion

Correlation

As shown in the results table, the correlation output image displays bright regions where the mask strongly matches the image content and darker regions where the match is poor. The brightest spots in the correlation output correspond to locations where the image neighborhood is most similar to the mask pattern. This demonstrates how correlations are useful as a template matching technique as it essentially measures the similarity between the mask and each local region of the image. The output serves as a similarity map, where intensity values represent the degree of correlation at each pixel location.

Smoothing

The smoothing results show that larger window sizes produce more aggressive smoothing and therefore a greater loss of fine details and stronger blurring effect. The Gaussian filter produces a crisper appearance compared to the average filter of the same size. This is because the Gaussian mask weighs central pixels heavier than surrounding pixels, creating a cleaner skew of values while still reducing noise with its positive weights. The averaging filter just applies equal weight to all the pixels which can cause overblurring which is more apparent in high-contrast regions of the image.

Median Filtering

At both noise levels, median filtering is significantly better than the gaussian filter in noise removal. The original image's features are generally retained in both mask sizes of 7 and 15 with median filtering on the noised images, while the gaussian filter blurs in the noise throughout the image. This difference in performance comes from the mathematical differences between

the median and the mean. The outliers from the salt-and-pepper noise added to the images influence the mean which causes the general noise in the gaussian smoothed image, while the median is resistant to outliers which allows it to find a good value from the uncorrupted pixels. This means that median filtering is preferred when there is impulse noise in the image causing outlier values.

Derivatives

The images show that the Prewitt and Sobel masks detect similar edges, with both finding reasonable edges in both partial derivative directions. The Sobel operator might be stated to produce slightly smoother results due to its increased weighting of the central row/column, which provides better noise suppression. The magnitude combines information from both directions and reveals all edges regardless of orientation. The Laplacian produced notably crisper and thinner edge lines compared to the gradient-based methods. This comes from the Laplacian's zero-crossing detection of where the gradient changes direction, marking more precise edge locations. The tradeoff of this is that the Laplacian is traditionally more sensitive to noise compared to the gradient, although that is not very noticeable in these results. The Laplacian was also easier to calculate as it only needed one mask to produce compared to two for the gradient, although it only holds magnitude information and not directional information like the gradient.

Sharpening

At k = 1 (standard unsharp masking), subtle sharpening is evident with enhanced edges and finer textures. As k increases, the sharpening effect becomes more apparent. Starting at k = 2 and up, the enhancement starts to become excessive and some pixels start to clip the brightness bounds of pixel values in the image. This shows the importance of selecting a good k value to not ruin the image while trying to enhance it.